Database Research at Bellcore

Editor: Amit P. Sheth
Bellcore, RRC 1J-210; 444 Hoes Lane; Piscataway, NJ 08854-4182

Bell Communications Research, Inc. (Bellcore) provides research and other technical support primarily to Ameritech, Bell Atlantic, BellSouth Corp., NYNEX, Pacific Bell, Southwestern Bell Corp., and U S WEST, Inc., as well as Cincinnati Bell, Inc., and the Southern New England Telephone Company. A large variety of database research and development activities are carried out at Bellcore. In this report, we describe eight selected projects that have significant research component and may interest SIGMOD Record readers. The projects discussed here address a number of issues, including architecture (Section 1), systems (Sections 2 and 3), query languages (Sections 3 and 4), data modeling (Section 3), database integration (Sections 5, 6, 7), user interface (Section 7), and databases in software engineering environment (Section 8).

1. Datacycle Architecture and Experimental Research Prototype

Participants: Gary Herman, Bill Mansfield, Tom Bowen, Dorothy Deluca, Gita Gopal, Prabhakar Krishnamurthy, Kasey Lee, Ravi Masand, Takako Matoba, John Raitz, Abel Weinrib.

The original goal of the Datacycle research project, which began in 1986, was to conceive a scalable, distributed transaction processing architecture that required minimal assumptions on application characteristics to permit extremely high throughputs. The approach was based on broadcasting database contents over very high bandwidth channels to a number of remote access nodes that performed predicate-based on-the-fly selection and aggregation operations.

As the project evolved, it became clear that the Datacycle architecture had merit for applications in which both generality (flexible access to data by different applications) and performance (throughput) were necessary. A major emphasis within Bellcore has been to create an application-independent view of corporate data and to reduce, and eventually eliminate, the dependency on application-specific databases and replicated data in telephone company operations and network systems. The Datacycle approach may allow implementation of multi-application databases that are currently beyond the state-of-the-art for commercial technology because it allows direct access to database contents based on any combination of attributes (there are no indexes in the Datacycle architecture), supports non-interfering access by multiple applications, maintains database consistency (uses optimistic concurrency control) and permits high volumes of interactions.

An experimental research prototype of the Datacycle architecture [Herman et al. 87, Herman and Gopal 89] has been implemented and is operational. The purpose of the research prototype [Bowen et al. 89] is to test the system concept through realization of critical system functionality, rather than through implementation of a full scale, full function system. The system is composed of two board types. A pump board broadcasts the contents of a memory based database to one or more access managers over a system backplane, or, alternatively, over an optical fiber using a 53 megabyte/second optical transmitter/receiver board. This allows access managers to be physically remote from the pump. The data filters used in the research prototype were designed using silicon compiler tools and fabricated in 2 micron CMOS [Lee and Herman 88]. They perform on-the-fly relational predicate evaluation and aggregation at up to 72 megabytes per second and support multi-tasking at the query level. Based on the performance characteristics of the research prototype, we have described application of the Datacycle approach to the future "intelligent network" [Bowen et al. 90]. By configuring an appropriate number of

pump and access manager boards, a system can, in principle, be engineered to meet requirements for storage volume, response time, and query throughput.

The current Datacycle research prototype acts as a database server on an Ethernet LAN and supports access from several types of hosts. Transaction management is based on the private workspace model. Existing software includes the query parser, compiler and assembler, resource management, transaction manager, and update certification and installation modules. The system supports ANSI SQL (a subset, at present) plus certain unusual extensions. We have implemented a fully content-addressable database application demonstration that permits retrieval of records based on any of a number of attributes. Arbitrary proximity queries (including those involving a "closest" primitive) are also supported. Wildcard queries, proximity queries, aggregation operations (sum, count, max, min) that would require full database scans in other approaches are executed with no performance penalty.

Beyond the functionality described in the original publication series, we have designed and plan to implement the following:

- Full ANSI Level 4 concurrency control (no phantom records); both record and predicate level certification of updates.
- Hot-spot management techniques.
- TP1 benchmark implementation (which is done quite differently in this architecture).
- Fault tolerance/recovery using off-site shadow pumps.
- Hardware-accelerated update certification.
- Fully-distributed architecture (distributed certification and pumps).
- Joins.
- Temporal and spatial extensions to the relational model, using the on-the-fly aggregate capabilities.
- Data placement strategies to improve hardware utilization.

[Herman et al. 87] G. Herman, et al., "The Datacycle Architecture for Very High Throughput Database Systems", *Proc. of ACM SIGMOD 87*, San Francisco, May 1987.

[Herman and Gopal 89] G. Herman and G. Gopal. "The Case for Orderly Sharing". *In High Performance Transaction Systems: Proc. of the 2nd Intl. Workshop*, D. Gawlick, M. Haynie, and A. Reuter (Eds)., Springer-Verlag Lecture Notes on Computer Science, 1989.

[Bowen et al. 89] T. Bowen et al., "A High Performance Distributed Database Using Transputers", Proceedings of the 1989 Meeting of the North American Transputers Users' Group, Durham, September 1989.

[Lee and Herman 88] K.C. Lee and G. Herman, "A High Performance Relational Data Filter". In Knowledgebases and Database Machines (*Proceedings of the 5th International Workshop on Database Machines*), Kluwer Academic Publishers, Boston, 1988.

[Bowen et al. 90] T. Bowen et al. "A Scalable Database Architecture for Network Services," *Proceedings of Intl. Switching Symposium*, Stockholm, May 1990.

2. Management of Interdependent Data

Participants: Marek Rusinkiewicz (visitor), Amit Sheth, Yungho Leu (summer student)

Like many large companies, Bellcore develops multiple databases that serve the needs of various application systems. The drive for vendor flexibility and the suitability of different environments for specific application requirements results in these databases being stored on multiple heterogeneous and autonomous systems. To allow interoperability among systems, Bellcore has developed an interoperability architecture

called the OSCA™ architecture [OSCA TA]. The data layer of this architecture supports sharing of data. One of the significant problems in managing databases in such an interoperable system is the maintenance of desired degree of consistency among related data items managed by different systems. In such environments, traditional correctness criterion of *one copy serializability* is too restrictive and may not be required. The issues we are addressing include:

Taxonomy of the Management of Interdependent Data

We characterize the problem according to the type of interdatabase dependency, data consistency criteria, and the degree of heterogeneity and local autonomy of the systems [Sheth, Rusinkiewicz 90]. The interdatabase dependencies include *structural dependency* (e.g., replication, partitioning, value constraints) and *control dependency* (e.g., derived data, primary-secondary copies). The mutual consistency criteria consist of temporal and spatial components. *Temporal requirements* may include immediate, eventual, or lagging consistency. *Spatial requirements* specify how far the related data may be allowed to diverge, by limiting the changes in the values of data or number of operations performed.

Specification of Interdatabase Dependencies

We are developing a specification language to describe relationships that exist between related data in an environment consisting of multiple autonomous database systems. The specifications include interdatabase dependencies and mutual consistency criteria. These specifications are stored in a federated schema and can be used to enforce the correctness of update transactions.

Multi-database Transaction Execution

The traditional transaction model requires that a transaction be executed atomically and its operations isolated from other concurrent transactions. For multidatabase transactions [Georgakopoulos et al. 90] involving multiple heterogeneous and autonomous DBMSs, these properties may be too restrictive or impossible to enforce. We are designing a multidatabase transaction execution environment that allows greater flexibility in the execution of transactions. In this environment a transaction may not require isolation and atomicity, and may also specify the required degree of data consistency.

OSCA is a trademark of Bell Communications Research, Inc.

[OSCA TA] The Bellcore OSCA Architecture, *Bellcore Technical Advisory TA-STS-000915*, Issue 2, July 1990.

[Georgakopoulos et al. 90] D. Georgakopoulos, M. Rusinkiewicz, A. Sheth, "On Serializability of Multidatabase Transaction through Forced Local Conflicts," *Technical Report UH-CS-90-20*, University of Houston, TX, June 1990.

[Sheth and Rusinkiewicz 90] A. Sheth and M. Rusinkiewicz, "Management of Interdependent Data: Specifying Dependency and Consistency Requirements," to appear in *Proc. of the Workshop on the Management of Replicated Data*, Houston, TX, November 1990.

3. Temporal Databases

Participants: Gene Wuu, Ramez Elmasri (visitor).

Temporal database technologies are fundamental to telecommunication network operations systems, especially to those systems providing the functions of network maintenance, planning, and provisioning. In these systems, network circuits and services are modeled as *complex objects* composed of interconnected network elements. The states of network objects and the relationships between them change over time. In order to support the functions of network maintenance, planning, and provisioning, these systems need to store not only the current state of the network, but also (1) the historical information and (2) the future *planned* evolutions of the network. The

temporal database project is addressing two issues discussed below.

Temporal Data Modeling and Query Language Design

Conventional data models are suitable for representing the current state of the application environment. Temporal information, on the other hand, adds the time dimension to the information space. A number of data models have been proposed to handle the temporal dimension. They are discussed mainly in the context of the relational model. In practice, the Entity-Relationship (ER) data model and its semantic extensions are the most widely used models for conceptual design. In [Elmasri and Wuu 90] we incorporate the temporal dimension into an extended ER model. The concept of lifespan for entities and relationships is included in the model. We also specify a temporal query and update language and discuss various temporal constraints supported by the model. Currently, we are studying techniques for incorporating the temporal dimension into object-oriented data models.

In addition to academic investigation in this area, a development team in Bellcore has been designing and implementing a temporal ER query language, SERQL, for production use in a telephone network provisioning system. The language is described in [Wuu]. In this paper we also discuss the concept of orthogonality between events and temporal objects, which was found to be very important in our planning applications.

Temporal Query Processing and Indexing Methods

There has been little research in the area of indexing temporal data and processing temporal operators efficiently. Existing storage techniques basically link or cluster the temporal versions of each individual object separately. These techniques cannot efficiently process temporal queries, such as "Retrieve the total number of employees who worked for the company on 1/1/1980.". We have developed a new index scheme, the TIME index [Elmasri et al. 90], which indexes temporal data items based on time intervals. TIME index leads directly to the desired versions that are valid during a specific period of time. We also describe algorithms to support temporal operators, such as *when*, *select*, *temporal joins*, and temporal aggregation functions. These temporal operators are typically used to specify queries such as "Find employees who have ever worked for a department headed by a person named Smith." Our simulation results show that the TIME index greatly improves the performance of these temporal queries relative to other temporal access structures.

[Elmasri et al. 90] R. Elmasri, G. Wuu, and Y.-J. Kim, "The TIME Index: An Access Structure for Temporal Data", *Proc. of the Sixteenth Very Large Data Bases Conference*, August 1990.

[Elmasri and Wuu 90] R. Elmasri and G. Wuu, "A Temporal Model and Language for ER Databases", *Proc. of the Sixth IEEE Data Engineering Conference*, February 1990.

[Wuu] G. Wuu, "SERQL: An ER Query Language Supporting Temporal Data Retrieval", *submitted for publication.*

4. Interpretable Databases

Participants: Tomasz Imielinski (consultant), Kumar Vadaparty (summer student), Shamim Naqvi, Yves Caseau, Madhur Kohli, Sam Epstein.

We are currently interested in representing interpretable data in a database system. In particular, one kind of interpretable data that we have considered is what we call an OR object, i.e., a set that represents many possible choices. The basic idea behind OR objects is to capture a certain notion of incompleteness: the object under study becomes more complete as decisions are made at different composition or abstraction levels. The decisions involve multiple choices (OR-trees) and constrained choices (AND-trees), with global constraints imposed on the completed object (e.g., no more than 20% of the total parts to be bought from a certain vendor, or no more than X% of traffic

through a node in the network).

Representing objects in different stages of completion calls for new functionalities of the query language. They include:

Local hypothetical queries – asking for consequences of a particular choice in terms of the completed object. For example, if we choose X now will the completed object have property P?

Global Hypothetical Queries – asking for consequences of a particular decision in terms of all global objects. For example, if we choose X, will every possible completion satisfy property P?

Multiple Abstraction – mechanisms to hide certain aspects of an object.

Additionally, one can talk about two different classes of queries:

Queries about a completion – these queries are concerned with the final "asymtotically" completed objects only.

Queries about the choices themselves – e.g., what are my next choices?

Besides these considerations, there are additional requirements on the DDL (Data Definition Language). The most important is the ability to define incomplete objects dynamically when the design choices at some level depend on the state of the database.

A formal understanding of such a database will provide answers to the following questions:

- what functionality is needed from a database to satisfy the queries that users need to ask?
- what can and cannot be expressed in the query language?
- what is the complexity of useful (large?) subsets of the query language?

5. Schema Integration

Participants: Amit Sheth, Howard Marcus, Ashok Savasere (student visitor), Sunit Gala (summer student in 1989).

In many large companies including Bellcore, different applications are supported by multiple heterogeneous and autonomous databases. A federated database system can provide uniform and integrated management of data in such environments [Sheth and Larson 90]. We are developing a schema integration software tool that is expected to be a critical component of a federated database toolkit [Sheth 88]. Such a toolkit is conceived to manage schemas and dictionary/directory information in a federated database system. The tool will assist a user in performing several closely related database design and integration tasks, including (a) integration of views when developing a large application, (b) identifying related data used by two or more applications, (c) developing a Corporate Logical Data Model by integrating schemas (also called application models) of applications throughout the company, and (d) integrating schemas of existing databases.

One of our research goals is to develop techniques to automatically integrate schemas whenever possible. However, the inability of current data models to adequately capture the semantics of the real world allows this goal to be only partially realized. This leads to a need to a two level approach consisting of manual integration and automatic integration [Sheth and Gala 89].

Manual integration involves activities which are performed with human input and reasoning. A Bellcore standard extended ER model is the underlying data model at this level. We have defined a meta-model of our ER model. A graphical interface allows creating and manipulating schemas that adhere to the integrity constraints of the meta-model. Forms are provided for entering and displaying dictionary information. We are defining a graphical query language for posing queries about the objects of the schema against the meta-model. The schema integrator can query the schemas using this language to determine the relationships among schema objects and can use a set of operators for manual integration of schema objects.

Schema objects include attributes (including relationships in ER model) and classes (e.g., entity types). It is not possible to automate the process of discovering how

attributes may be related. Hence the specification of the attribute relationships, including the superset and subset relationships, involves human inputs [Sheth and Gala 89]. The tool guides the integrator through this process.

For automatic integration, the schemas are converted to a formal data model called CANDIDE. CANDIDE provides the classification paradigm of KL-ONE systems. Classification provides formal reasoning to determine the superclass-subclass relationships among classes. We have implemented additional functionality in CANDIDE to determine a complete set of class relationships such as the equivalence, overlap and disjointness. These can be used to automatically identify a good percentage of class relationships and develop an integrated schema consisting of class definitions of the schemas being integrated.

Our approach identifies those operations that can be automated and those that are essentially manual. We plan to determine, using real life schemas, if a significant portion of the schema integration can be automated by using formal reasoning.

[Sheth 88] A. Sheth, "Building Federated Database Systems," *Distributed Computing Technical Committee Newsletter*, Vol. 10, No. 2, November 1988.

[Sheth and Gala 89] Attribute Relationships: An Impediment in Automating Schema Integration, *Proc. of the Workshop on Heterogeneous Database Systems*, Chicago, December 1989.

[Sheth and Larson 90] A. Sheth and J. Larson, "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," to appear in *ACM Computing Surveys*, September 1990.

6. Heterogeneous Distributed Database Testbed

Participants: Arbee L.P. Chen, Fred Cohen, Prabhakar Krishnamurthy, Gomer Thomas, Ray-John Wang.

The Heterogeneous Distributed Database Testbed project was started at Bellcore to experiment with heterogeneous distributed database technologies and to prototype software with capabilities not likely to be available in vendor-supplied systems. The project was intended to investigate the feasibility of the principles proposed in the data layer of the OSCA architecture (see Section 2), and provide feedback to the OSCA architects and the developers about the principles.

During the first phase of the project, standardized remote database access interfaces were developed for a number of commercial database management systems used at Bellcore. Experiments were done to gauge the performance of applications accessing databases remotely and executing in a client/server configuration [Krishnamurthy 90].

The second phase of the project was the development of distributed query processing capability. A set of operators were defined for providing an integrated global schema for accessing heterogeneous databases in a distribution transparent manner [Chen 90c]. A distributed query manager for decomposing global queries was developed by porting the OMNIBASE query manager [Rusinkiewicz 88]. A research prototype redundant data manager was also developed. Updates to databases in the testbed can only be done by invoking pre-defined procedures. The purpose of the redundant data manager is to automatically propagate updates to inter-related data stored in different databases, upon the invocation of an update procedure by a user (see Section 2).

Currently, efforts are being made to improve execution of global queries on heterogeneous databases. Algorithms for optimization of outerjoin operation, which is a useful operator for integrating local schemas have been proposed [Chen 90b]. Conditions under which a distributed query can be transformed and executed locally have been identified [Chen 90a]. Experiments are also being performed to determine good query decomposition and optimization strategies for heterogeneous databases.

[Chen 90a] Chen, A.L.P., "A Localized Approach to Distributed Query Processing," *Proc. Intl. Conf. on Extending Data Base Technology*, 1990.

[Chen 90b] Chen, A.L.P., "Outerjoin Optimization in Multidatabase Systems," *Proc. Intl. Symposium on Databases in Parallel and Distributed Systems*, 1990.

[Chen 90c] Chen, A.L.P., "Relational Schema Integration and Distributed Query Optimization," accepted by *IEEE TENCON*, 1990.

[Krishnamurthy 90] Krishnamurthy, P., "Performance Analysis of Remote Database Access", *Proc. The Twenty-Third Annual Hawaii Intl. Conf. on System Sciences*, 1990.

[Rusinkiewicz 88] Rusinkiewicz, M. et al, "Query Processing in OMNIBASE - A Loosely Coupled Multi-Database System", *Technical Report*, University of Houston, Feb. 1988.

7. The Cheyenne Project

Participants: George Collier, Aita Salasoo, Fred Cohen, Mohan Pillalamarri, Gerry Grate.

Study of telephone operating company planners and engineers in the US has helped us to identify three major ways in which telecommunication network views may be improved. Information which currently exists in separate databases or paper records may be presented in an **integrated** way to users of that information. Information may be shown in a way that corresponds to *meaningful* units or objects, rather than to structures convenient to the computer or file storage system. An interface to network information may seem *intelligent* to a user, if it provides information that corresponds to the task context, specific level of detail, and display type desired by the user. We have developed a prototype of an intelligent, integrated, and meaningful software system that combines new graphical interface technologies with object-oriented database and knowledge representation technologies. It is the first attempt we know of that combines state-of-the-art object-oriented databases, user-centered design, and network based data integration techniques to serve telephone company network operations.

Data from multiple existing sources are extracted as needed by the user and combined into graphical views. This happens without the user having to access the separate data sources explicitly, or to know the source of the requested information. A User Interface Management System is used to keep track of the relation between a presentation of information and the underlying data structure. This allows many more manipulations of network data than the originating data sources. For example, maps with cable records, copper assignment data, and electronic inventory data today constitute separate databases for engineers. In our prototype software system, information from any combination of these sources can be obtained easily to solve design, maintenance, and network service provisioning problems. This is done by pointing and clicking on relevant visible objects on the computer screen, or by selection from menus, or a powerful command language. This integration is the keystone, we believe, to developing the powerful engineering applications of the future.

[Collier et al. 90] G. Collier, L. Walko, and P. Cannata, "Pluralism in System Integration: Intelligent LEIM," *Proc. of the First Intl. Conf. on System Integration*, Morristown, NJ, April 1990.

[Salasoo] A. Salasoo, "Towards Usable Icon Sets: A Case Study from Telecommunications Engineering, *Proc. of 34th Annual Meeting of Human Factors Society*, Orlando, FL, October 1990.

[Salasoo and Collier 90] A. Salasoo and G. Collier, "Integrated Network Views for Enhanced Operations," *submitted for publication.*

8. Bellcore Integrated Software Environment

Participants: Reva Leung, Gomer Thomas, Chit Chung, Maryam Asdjodi, Abe Shliferstein, Suresh Subramanian, Victor Gregg, Todd Moyer, Bob Mitchell, Josh Nabozny.

The ultimate goal of the Integrated Software Environment (ISE) project is to provide for internal use at Bellcore a fully integrated environment to support all activities in the software development life cycle and its associated work environment. These activities include planning, requirements analysis, design, coding and debugging, quality assurance, customer training, project management, preparing reports and presentations, scheduling meetings, filling out forms, and looking up technical references.

There are three key types of integration in such an environment – data integration, process integration, and user interface integration.

Data integration means that users and software tools have a shared understanding of the information being used – what entities or objects exist, how they are structured, how they are interrelated, and where to find them (also see Section 5).

Process integration means that the user typically deals with processes or activities, rather than with individual software tools. When an activity involves multiple tools, there is a smooth flow from one to the next, without any need to specify each step.

User interface integration means the user has a uniform interface for initiating different activities, and the user sees a consistent interface while performing different activities.

>From a data integration perspective, the first key task in the ISE project is to develop an information model (or schema). This serves the dual roles of initially helping to define the information management requirements and later helping users and tools understand the information structure and interrelationships. For this an ER modeling approach is being used, with a number of object-oriented extensions such as subtypes/supertypes (with multiple inheritance) and descriptions of allowable operations on entity types.

The second key task is implementation – specifying and providing appropriate mechanisms for managing the diverse types of information in the environment. These must satisfy the diverse requirements for ease of access, security, versioning, performance, etc.

It is expected that implementation will proceed in three phases, which are likely to overlap chronologically to some extent. Phase 1 is concentrating on those types of information that are fairly loosely interrelated and that are not closely tied to the diverse target environments on which Bellcore-developed software runs – free-form text requirements and design documents, various kinds of reports, presentation foils, administrative forms or policies, etc. The emphasis is on providing an integrated set of state-of-the-art tools for electronically creating, distributing, storing, and retrieving these types of information. It is expected that the tools will include desktop publishing software, mailer and bulletin board software, and document retrieval software, perhaps augmented with some form of information encyclopedia. Phase 2 will extend coverage to most other types of information in the environment, including project management, software configuration management, and testing information. In this phase there will be more emphasis on tools accessing the information, as opposed to interactive users. An information encyclopedia will play a more central role. Phase 3 will concentrate on encompassing all the software environment information in a logically unified information store and generally smoothing out the seams. The information store may be an integral part of an overall tool integration platform, or it may be a separate object management system of some kind.